



Europäisches Patentamt

European Patent Office

Office européen des brevets

⑪ Publication number:

0 325 238
A2

⑫

EUROPEAN PATENT APPLICATION

⑬ Application number: 89100825.2

⑭ Int. Cl. 4 G07F 7/10

⑮ Date of filing: 18.01.89

⑯ Priority: 19.01.88 US 145402

⑰ Applicant: YEDA RESEARCH AND
DEVELOPMENT COMPANY, LIMITED
P.O. Box 95
Rehovot 76100(IL)

⑰ Date of publication of application:
26.07.89 Bulletin 89/30

⑲ Inventor: Shamir, Adi
43C Gordon Street
Rehovot(IL)

⑳ Designated Contracting States:
AT BE CH DE ES FR GB GR IT LI LU NL SE

㉑ Representative: Patentanwälte Grünecker,
Kinkeldey, Stockmair & Partner
Maximilianstrasse 68
D-8000 München 22(DE)

㉒ Improved variants of the flat-shamir identification and signature scheme.

㉓ A method of proving the identity of an entity comprising the steps of establishing a public key for the entity consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k ; computing a private key for the entity consisting of k numbers s_1, \dots, s_k satisfying $s_j^d v_j = 1 \pmod{n}$ for all $1 \leq j \leq k$, where d is a universally known constant larger than 1; transmitting from the entity to a verifier $x = r^d \pmod{n}$ where r is a random number in the range $0 < r < n$; transmitting from the verifier to the entity a sequence of k random numbers e_1, \dots, e_k in the range $0 < e_j < d$; transmitting from the entity to the verifier the value

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n};$$

and verifying the identity of the entity by checking that

$$x = y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}.$$

EP 0 325 238 A2

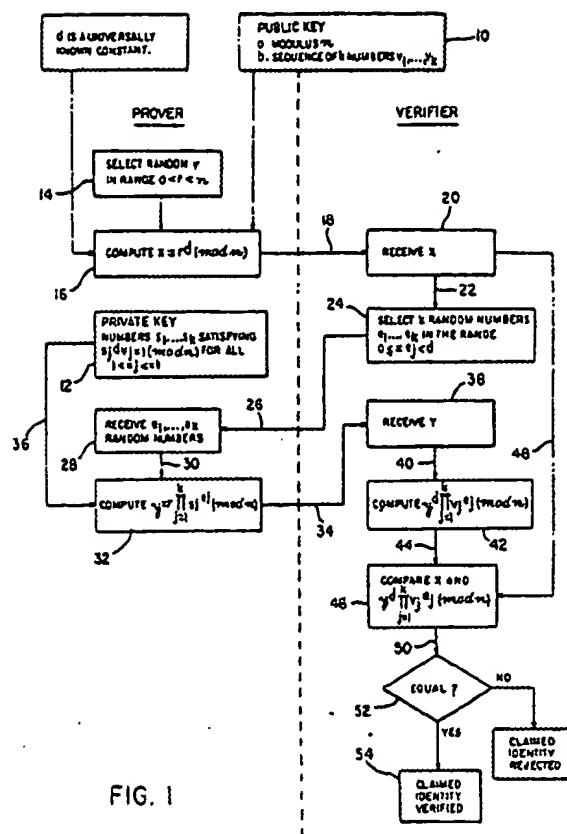


FIG. 1

IMPROVED VARIANTS OF THE FIAT-SHAMIR IDENTIFICATION AND SIGNATURE SCHEME

FIELD OF INVENTION

The present invention relates to a method and apparatus for implementing an identification and signature scheme and, more particularly, relates to an improvement to the method and apparatus disclosed and claimed in, and is a continuation-in-part of, U.S. Patent Application Serial No. 08/883,247 filed July 9, 1986 in the names of Adi Shamir and Amos Fiat.

SUMMARY OF INVENTION

In the aforementioned parent patent application (to which we refer henceforth as the original Fiat-Shamir scheme), a method and apparatus are disclosed and claimed which enable an entity to generate proofs of identity and signatures of messages that everyone can verify but no one can forge. Compared to other schemes of this type (such as the RSA), the Fiat-Shamir scheme is faster by one to two orders of magnitude. In the present invention disclosed and claimed in this patent application, two improved variants of the Fiat-Shamir scheme are described which are even faster. One variant uses small public values to optimize the verification procedure, and the other variant uses small secret values to optimize the generation procedure. To keep these variants secure and to fully exploit their improved performance, some of the details of the original method and apparatus disclosed and claimed in the aforementioned parent patent application have been modified. In particular, the public key of an entity is no longer computed as a function of its identity I , the quadratic expressions are generalized to d -th powers, and the execution of the protocol is usually not iterated, but may be.

Other and further objects and advantages of the present invention will be evident from the following detailed description taken in conjunction with the drawings.

26

DESCRIPTION OF THE DRAWINGS

Figure 1 is a schematic showing the novel method and apparatus for proving identity; and

Figure 2 is a schematic showing the novel method and apparatus for generating and verifying a signature to a message.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

35

Referring now to the drawings, the basic structure of the new and improved variants of the method and apparatus of the aforementioned parent patent application is illustrated in Figure 1 for a proof of identity. As shown, an entity either selects or is given a public key, see block 10, consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k . This public key can be stored in a public key directory, or sent to the verifier along with a trusted center's signature on it to establish its authenticity. The entity computes a corresponding private key, block 12, consisting of numbers s_1, \dots, s_k satisfying $s_j^d v_j = 1 \pmod{n}$ for all $1 \leq j \leq k$, where d is some universally known constant larger than 1.

Accordingly, the identification technique proceeds as follows.

To prove his, hers or its identity, the entity chooses a random r in the range $0 < r < n$, block 14, and sends $x = r^d \pmod{n}$ to the verifier, block 16 and line 18, where it is received by the verifier, block 20. Upon receipt of x , line 22, the verifier chooses k random numbers e_1, \dots, e_k in the range $0 \leq e_j < d$, block 24, and sends them to the prover, line 26. The prover, in response to receipt of e_1, \dots, e_k , block 28, computes and sends

50

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n}$$

to the verifier, block 32, line 34. Line 36 sends the s_j values from block 12 to block 32. The verifier upon

receipt of y value in block 38, via line 40, computes in block 42 the value

$$y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}.$$

5

This value is sent via line 44 to block 46 where it is compared with the value x , via line 48. The result is passed by line 50 to block 52 where the judgment is made. The verifier accepts the proof of identity in block 54 if

10

$$x = y^d \prod_{j=1}^k v_j^{e_j} \pmod{n},$$

15

and rejects the proof of identity if this equality does not hold.

The signature technique of the present invention is schematically portrayed in Figure 2. As shown, the identification scheme of Figure 1 is turned into a signature scheme by using some publicly known 20 cryptographically strong pseudo random function f which maps its inputs into a sequence of k numbers $e = e_1, \dots, e_k$ in the range $0 \leq e_j < d$, block 70. To generate a signature for message m according to the present invention, first, a random r in the range $0 < r < n$ is chosen. Then, $e = f(r^d \pmod{n}, m)$ is computed in block 72. Next

25

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n}$$

30

is computed in block 74. The signature of m consists of e and y . These values are either stored or sent to the verifier where they are received in block 76.

To verify the stored or transmitted signature, the verifier computes

35

$$f \left(y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}, m \right),$$

40

block 78. These values are compared with e , block 80. If equal, block 82, the signature is accepted as genuine and, if not, the signature is rejected as a forgery.

45

The best known attack on these identification and signature techniques has a probability of success of d^{-k} per attempt. To make the interactive identification protocol secure, it usually suffices to choose d^k larger than or equal to 2^{20} since a cheater has only one chance to forge a proof of identity. To make the non-interactive signature technique secure against repeated attempts to forge a signature, d^k should be at least 2^{54} . This can be achieved either by using sufficiently large values of d and k , or by iterating the executions of these schemes t times and making sure that $d^{kt} \geq 2^{54}$. In most applications, the first option is recommended since it minimizes both the number of modular multiplications and the number of communicated bits.

50

In the first variant of the present invention, small v_i values are used. In the preferred embodiment of this variant, d is 2 or 3, and all the entities choose the numbers v_i as the first k primes $v_1 = 2, v_2 = 3, v_3 = 5$, etc. Since the v_i are universal, only the modulus n has to be specified in the public-key directory. In a typical implementation of this variant with $k = 64$ numbers and 512 bit modulus, the size of each entry in the public-key directory is 64 bytes, and the size of each signature is 72 bytes, which are comparable with those of the RSA signature scheme. The size of the private key is about 4 kilobytes, but since each entity has to store only one such file, it can fit into almost any microcomputer based device (with the possible exception of a smart card). With optimized implementations, it is possible to generate proofs of identity in less than 10 modular multiplications and to generate signatures in less than 30 modular multiplications, which is the

same as in the original Fiat-Shamir scheme. However, since multiplications by the small v_j values (most of which fit in a single byte) have negligible complexity, the verification of identities and signatures in this variant requires only 1 or 2 full-size modular multiplications. This is an order of magnitude faster than in the original Fiat-Shamir scheme, and is expected to take only tens of milliseconds on a standard microprocessor and less than a millisecond on a mainframe computer. The high efficiency of verification can be crucial when a central computer has to verify many access requests or signed documents generated in thousands of terminals, or when a remotely controlled robot has to verify in real time a stream of signed instructions generated by a more powerful central computer.

The choice of exponent d depends on the relative importance of efficiency and convenience. When $d=2$, the scheme requires the fewest number of modular multiplications, but the square roots of the v_j do not always exist. It is thus necessary to modify the scheme in one of the following ways.

1. Each entity can choose its own set of small v_j values, making sure that all of them are quadratic residues modulo the entity's n . These v_j 's should be published in full in the public-key directory.
2. Use a longer list of standard v_j values and each entity can choose a subset of k values which are quadratic residues modulo its own n (when n is the product of two primes, the universal list should contain about $4k$ numbers). The entity's chosen subset should be specified in the public-key directory or sent as part of the signature.

3. Each entity is allowed to modify the standard v_j which are quadratic non-residues. A particularly simple way to achieve this is to pick a modulus $n = pg$ where $p \equiv 3 \pmod{8}$ and $g \equiv 7 \pmod{8}$, since then exactly one of $v_j, -v_j, 2v_j, -2v_j$ is a quadratic residue mod n for any v_j . The appropriate variant of each v_j can be specified in the public-key directory, sent as part of the signature, or deduced by the verifier himself during the verification of given signatures.

Alternatively, the entities can use $d=3$ and avoid all these complications. If the totient function of n is not divisible by 3, then any v_j value has a cubic root mod n . However, this choice of d requires an additional modular multiplication during the generation and the verification of signatures.

The choice of the v_j as the first k primes in the preferred embodiment is based on the fact that large values can make the scheme less efficient, and multiplicatively related values can make the scheme less secure. However, the v_j can be chosen as any other sequence of k numbers.

As described in the parent Fiat-Shamir application, there are many possible tradeoffs between the size of the keys, the number of communicated bits, and the number of modular multiplications. All the optimization ideas described in the parent patent application are equally applicable to this variant.

In a second variant of the present invention, small s_j values are used. In this variant, the entities can choose their own moduli n , or can use a universal modulus n published by a trusted center (the first option is more secure, while the second option reduces the size of the public key). Each entity chooses a sequence of small secret numbers s_1, \dots, s_k , and computes the corresponding public numbers v_1, \dots, v_k as

$$v_j = \frac{1}{s_j^d} \pmod{n}$$

(note that this computation can be carried out even when the factorization of n is unknown). Each s_j should be at least 64 bits long to protect it from exhaustive search attacks, and the exponent d should be large enough to guarantee that $s_j^d > n$ (e.g., when $|n| = 512$ and $|s_j| = 64$, d should be at least 16 to guarantee sufficient wraparound). The computed values of v_1, \dots, v_k (and the entity's modulus n , if applicable) are placed in the public key directory. The actual generation and verification of proofs of identity and signatures is carried out in the way specified above. The small values of the s_j have two beneficial effects.

1. The size of the private key is much smaller than in the above version of the signature technique. To make $d^k = 2^{64}$ for $d=16$, it suffices to use $k=16$ instead of $k=64$ s_j values, and the size of each s_j is 64 bits instead of 512 bits. The total size of the private key is thus reduced by a factor of 32, from 4096 bytes to 128 bytes.

2. Signature generation becomes about 3 times faster since the multiplication by the smaller s_j values is more efficient.

Both properties are highly desirable when the signature-generation process has to be carried out in a smart card with severe limitations on the available memory and computing power.

Known components can be used for the apparatus of the present invention as portrayed in Figure 1 and Figure 2, the means to carry out the several steps of the process being apparent to those skilled in the art.

Although the invention has been shown and described in terms of a specific preferred embodiment and variants, changes and modifications are possible which do not depart from the spirit, scope or contemplation of the inventive concepts disclosed herein. Such are deemed to fall within the purview of the invention as claimed.

Claims

10. 1. A method of proving the identity of an entity comprising the steps of
 - a) establishing a public key for the entity consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k ;
 - b) computing a private key for the entity consisting of k numbers s_1, \dots, s_k satisfying $s_j^d v_j = 1 \pmod{n}$
15. for all $1 \leq j \leq k$, where d is a universally known constant larger than 1;
 - c) transmitting from the entity to a verifier $x = r^d \pmod{n}$ where r is a random number in the range $0 < r < n$;
 - d) transmitting from the verifier to the entity a sequence of k random numbers e_1, \dots, e_k in the range $0 \leq e_j < d$;
20. e) transmitting from the entity to the verifier the value

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n};$$

25. f) verifying the identity of the entity by checking that

$$x = y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}.$$

30. 2. A method as claimed in claim 1 wherein steps c, d, e and f are repeated t times where $t \geq 1$.
3. A method as claimed in claim 1 wherein the v_j 's are the same for all the entities.
4. A method as claimed in claim 1 wherein the v_j 's are the first k prime numbers.
5. A method as claimed in claim 1 including the further step of storing the modulus n and the k numbers v_1, \dots, v_k in a public key directory.
6. A method as claimed in claim 1 including the further step of transmitting the modulus n and the k numbers v_1, \dots, v_k from the entity to the verifier along with a trusted center's signature on this public key.
40. 7. A method as claimed in claim 3 including the further step of storing the modulus n in a public key directory.
8. A method as claimed in claim 3 including the further step of transmitting the modulus n from the entity to the verifier along with a trusted center's signature on this public key.
45. 9. A method as claimed in claim 1 wherein $d^k \geq 2^{20}$.
10. A method as claimed in claim 1 wherein d is either 2 or 3.
11. A method as claimed in claim 1 wherein k is at least 20.
12. A method as claimed in claim 1 wherein n is at least 512 bits long.
50. 13. A method as claimed in claim 1 wherein the secret key s_1, \dots, s_k is chosen first and each v_j is then computed as

$$v_j = \frac{1}{s_j^d} \pmod{n}.$$

55

14. A method as claimed in claim 13 wherein all the entities use the same modulus n which is chosen by a trusted center that keeps its factorization secret.

15. A method as claimed in claim 13 wherein the s_j are chosen as random b bit numbers with $b < |n|$ and $s_j > n$ for all $1 \leq j \leq k$.

16. A method of generating a signature for a message m comprising the steps of
 a) establishing a public key for the entity consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k ;
 b) computing a private key for the entity consisting of k numbers s_1, \dots, s_k satisfying $s_i^{d_j} = 1 \pmod{n}$ for all $1 \leq i \leq k$, where d is a universally known constant larger than 1;
 c) agreeing on a common cryptographically strong pseudo random function f which maps its inputs into a sequence of k numbers $e = e_1, \dots, e_k$ in the range $0 \leq e_j < d$;
 d) choosing a random number r in the range $0 < r < n$ and computing $\underline{e} = f(r^d \pmod{n}, m)$;
 e) computing

$$15 \quad y = r \prod_{j=1}^k s_j^{e_j} \pmod{n};$$

20. f) transmitting or storing \underline{e} and y as the entity's signature on m .

17. A method of verifying a signature \underline{e}, y for a message m comprising the steps of
 a) computing

$$25 \quad f \left(y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}, m \right);$$

30. b) accepting the signature as valid if and only if these values are equal to \underline{e} .

18. A method as claimed in claim 16 wherein $t \geq 1$ random numbers r_1, \dots, r_t are chosen in the range $0 < r_i < n$, $\underline{e} = e_1, \dots, e_t$ numbers in the range $0 \leq e_j < d$ are computed as $\underline{e} = f(r_1^d \pmod{n}, \dots, r_t^d \pmod{n}, m)$, the values y_1, \dots, y_t are computed as

35

$$y_j = r_j \cdot \prod_{j=1}^k s_j^{e_j} \pmod{n},$$

40. and $\underline{e}, y_1, \dots, y_t$ are transmitted or stored as the entity's signature on m .

19. A method as claimed in claim 17 for verifying a signature $\underline{e}, y_1, \dots, y_t$ generated by the method of claim 18 for a message m in which the signature is accepted as verified if, and only if,

$$45 \quad \underline{e} = f \left(y_1^d \prod_{j=1}^k v_j^{e_j} \pmod{n}, \dots, y_t^d \prod_{j=1}^k v_j^{e_j} \pmod{n}, m \right).$$

50. 20. A method as claimed in claim 18 wherein the v_j 's are the same for all the entities.

21. A method as claimed in claim 18 wherein the v_j 's are the first k prime numbers.

22. A method as claimed in claim 18 including the further step of storing the modulus n and the k numbers v_1, \dots, v_k in a public key directory.

23. A method as claimed in claim 18 including the further step of incorporating the modulus n , the k numbers v_1, \dots, v_k , and a trusted center's signature on these values as part of the signature of the message m .

55. 24. A method as claimed in claim 20 including the further step of storing the modulus n in a public key directory.

25. A method as claimed in claim 20 including the further step of incorporating the modulus n and a trusted center's signature on it as part of the signature of the message m.
26. A method as claimed in claim 18 wherein $d \geq 2^{64}$.
27. A method as claimed in claim 18 wherein d is either 2 or 3.
28. A method as claimed in claim 18 wherein k is at least 64.
29. A method as claimed in claim 18 wherein n is at least 512 bits long.
30. A method as claimed in claim 18 wherein the secret key s_1, \dots, s_k is chosen first and each v_j is then computed as

$$v_j = \frac{1}{s_j^d} \pmod{n}.$$

31. A method as claimed in claim 30 wherein all the entities use the same modulus n which is chosen by a trusted center that keeps its factorization secret.
32. A method as claimed in claim 30 wherein the s_j are chosen as random b bit numbers with $b < |n|$ and $s_j^d > n$ for all $1 \leq j \leq k$.
33. Apparatus for proving the identity of an entity comprising
- a) means for establishing a public key for the entity consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k ;
 - b) means for computing a private key for the entity consisting of k numbers s_1, \dots, s_k satisfying $s_j^d v_j = 1 \pmod{n}$ for all $1 \leq j \leq k$, where d is a universally known constant larger than 1;
 - c) means for transmitting from the entity to a verifier $x = r^d \pmod{n}$ where r is a random number in the range $0 < r < n$;
 - d) means for transmitting from the verifier to the entity a sequence of k random numbers e_1, \dots, e_k in the range $0 \leq e_j < d$;
 - e) means for transmitting from the entity to the verifier the value

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n};$$

- f) means for verifying the identity of the entity by checking that

$$x = y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}.$$

34. Apparatus as claimed in claim 33 including means for repeating t times where $t \geq 1$, the transmitting from the entity to a verifier $x = r^d \pmod{n}$ where r is a random number in the range $0 < r < n$, the transmitting from the verifier to the entity of a sequence of k random numbers e_1, \dots, e_k in the range $0 \leq e_j < d$, the transmitting from the entity to the verifier the value

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n},$$

- and the verifying the identity.
35. Apparatus as claimed in claim 33 wherein the v_j 's are the same for all the entities.
36. Apparatus as claimed in claim 33 wherein the v_j 's are the first k prime numbers.
37. Apparatus as claimed in claim 33 including means for storing the modulus n and the k numbers v_1, \dots, v_k in a public key directory.
38. Apparatus as claimed in claim 33 including means for transmitting the modulus n and the k numbers v_1, \dots, v_k from the entity to the verifier along with a trusted center's signature on this public key.

39. Apparatus as claimed in claim 35 including means for storing the modulus n in a public key directory.

40. Apparatus as claimed in claim 35 including means for transmitting the modulus n from the entity to the verifier along with a trusted center's signature on this public key.

5 41. Apparatus as claimed in claim 33 wherein $d \geq 2^20$.

42. Apparatus as claimed in claim 33 wherein d is either 2 or 3.

43. Apparatus as claimed in claim 33 wherein k is at least 20.

44. Apparatus as claimed in claim 33 wherein n is at least 512 bits long.

10 45. Apparatus as claimed in claim 33 wherein the secret key s_1, \dots, s_k is chosen first and each v_j is then computed as

$$v_j = \frac{1}{s_j^d} \pmod{n}.$$

15

46. Apparatus as claimed in claim 45 wherein all the entities use the same modulus n which is chosen by a trusted center that keeps its factorization secret.

20 47. Apparatus as claimed in claim 45 wherein the s_j are chosen as random b bit numbers with $b < |n|$ and $s_j^d > n$ for all $1 \leq j \leq k$.

48. Apparatus for generating a signature for a message m comprising

a) means for establishing a public key for the entity consisting of a modulus n which is the product of at least two prime numbers, and a sequence of k numbers v_1, \dots, v_k ;

25 b) means for computing a private key for the entity consisting of k numbers s_1, \dots, s_k satisfying $s_j^d v_j = 1 \pmod{n}$ for all $1 \leq j \leq k$, where d is a universally known constant larger than 1;

c) means for agreeing on a common cryptographically strong pseudo random function f which maps its inputs into a sequence of k numbers $e = e_1, \dots, e_k$ in the range $0 \leq e_j < d$;

d) means for choosing a random number r in the range $0 < r < n$ and computing $e = f(r^d \pmod{n}, m)$;

30 e) means for computing

$$y = r \prod_{j=1}^k s_j^{e_j} \pmod{n};$$

35

f) means for transmitting or storing e and y as the entity's signature on m.

49. Apparatus for verifying a signature e, y for a message m comprising

a) means for computing

$$f \left(y^d \prod_{j=1}^k v_j^{e_j} \pmod{n}, m \right);$$

45

b) means for accepting the signature as valid if and only if these values are equal to e .

50. Apparatus as claimed in claim 48 wherein $t \geq 1$ random numbers r_1, \dots, r_t are chosen in the range $0 < r_i < n$, $e = e_1, \dots, e_t$ numbers in the range $0 \leq e_j < d$ are computed as $e = f(r_1^d \pmod{n}, \dots, r_t^d \pmod{n}, m)$, t values y_1, \dots, y_t are computed as

$$y_i = r_i \cdot \prod_{j=1}^k s_j^{e_j} \pmod{n},$$

55

and e, y_1, \dots, y_t are transmitted or stored as the entity's signature on m.

51. Apparatus as claimed in claim 49 for verifying a signature e, y_1, \dots, y_t generated by the apparatus of claim 50 for a message m in which the signature is accepted as verified if, and only if,

$$e=f \left(y_1^d \prod_{j=1}^k v_j e_j^t \pmod{n}, \dots, y_k^d \prod_{j=1}^k v_j e_j^t \pmod{n}, m \right).$$

- 5 52. Apparatus as claimed in claim 50 wherein the v_j 's are the same for all the entities.
 53. Apparatus as claimed in claim 50 wherein the v_j 's are the first k prime numbers.
 54. Apparatus as claimed in claim 50 including means for storing the modulus n and the k numbers v_1, \dots, v_k in a public key directory.
 10 55. Apparatus as claimed in claim 50 including means for incorporating the modulus n , the k numbers v_1, \dots, v_k , and a trusted center's signature on these values as part of the signature of the message m .
 56. Apparatus as claimed in claim 52 including means for storing the modulus n in a public key directory.
 57. Apparatus as claimed in claim 52 including means for incorporating the modulus n and a trusted center's signature on it as part of the signature of the message m .
 15 58. Apparatus as claimed in claim 50 wherein $d \geq 2^{64}$.
 59. Apparatus as claimed in claim 50 wherein d is either 2 or 3.
 60. Apparatus as claimed in claim 50 wherein k is at least 64.
 61. Apparatus as claimed in claim 50 wherein n is at least 512 bits long.
 20 62. Apparatus as claimed in claim 50 wherein the secret key s_1, \dots, s_k is chosen first and each v_j is then computed as

$$v_j = \frac{1}{s_j^d} \pmod{n}.$$

- 25 63. Apparatus as claimed in claim 62 wherein all the entities use the same modulus n which is chosen by a trusted center that keeps its factorization secret.
 30 64. Apparatus as claimed in claim 62 wherein the s_j are chosen as random b bit numbers with $b < |n|$ and $s_j^d > n$ for all $1 \leq j \leq k$.

35

40

45

50

55

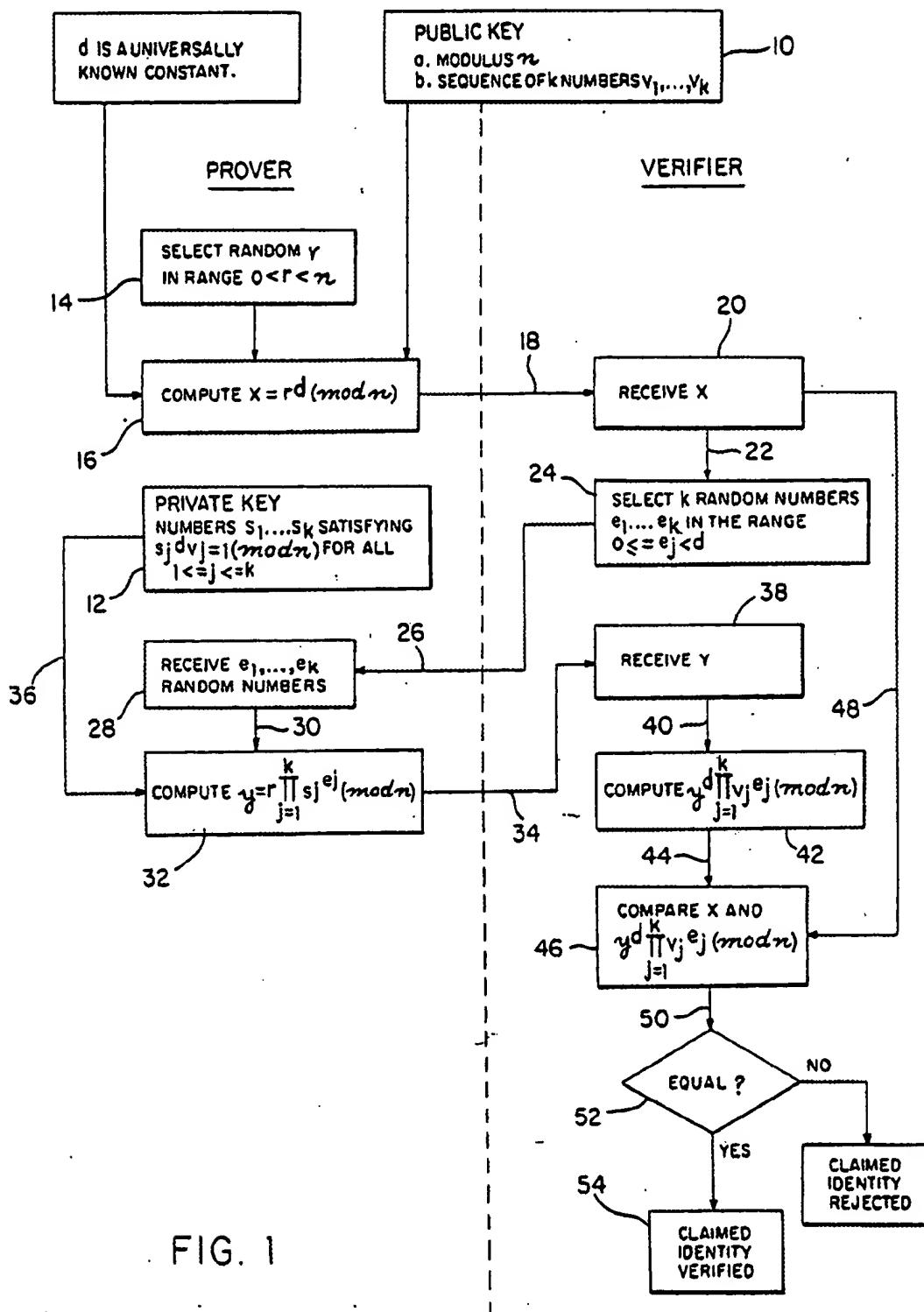


FIG. 1

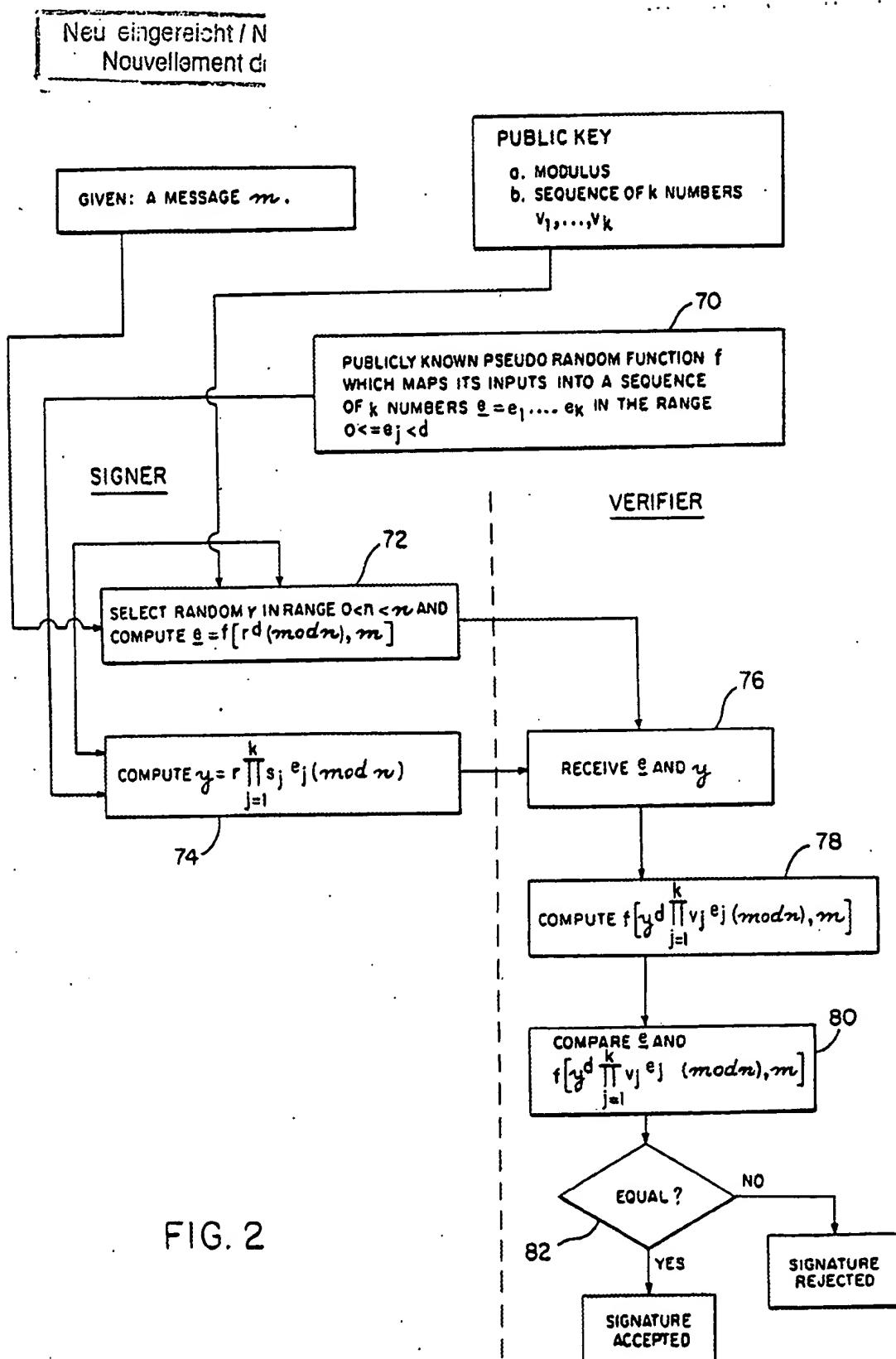


FIG. 2